INSTITUTO POLITÉCNICO
DO CÁVADO E DO AVE

Escola superior de Tecnologia

Processamento de linguagens

Autores:
Diogo Bernardo ***21144***
João Ribeiro ***23795***
José Fangueiro ***23515***

Docentes:
Prof. Óscar Ribeiro

Submissão do projeto para
o curso de Engenharia de Sistemas Informáticos

ESI

June 4, 2023

IPCA

Resumo

Engenharia de Sistemas Informáticos

Processamento de linguagens

por Diogo Bernardo **21144**
João Ribeiro **23795**
José Fangueiro **23515**

Trabalho em cargo da disciplina de Processamento de linguagens.

Contents

Resumo **iii**

1 Introdução **1**
1.1 Problema que foi resolvido no trabalho prático 1
1.2 Importância e contexto do problema 1
1.3 Objetivo geral do trabalho 2

2 Metodologia **3**
2.1 Abordagem seguida para resolver o problema 3
2.2 Descrição de como a leitura do arquivo de texto foi realizada e como os comandos foram interpretados 4
2.3 Estrutura da Árvore de Sintaxe Abstrata (AST) utilizada na implementação 4

3 Funcionalidades implementadas **5**
3.1 Listagem e explicação das funcionalidades principais da linguagem de expressões aritméticas implementada 5
3.2 Funcionalidades adicionais implementadas além dos requisitos mínimos do enunciado 5
3.3 Explicação de como as funcionalidades foram integradas à linguagem . 5

4 Desafios encontrados **7**
4.1 Principais desafios enfrentados durante o desenvolvimento 7

5 Resultados **9**
5.1 Apresentação dos resultados obtidos com a implementação da ferramenta 9
5.2 Explicação de como o programa foi capaz de gerar um arquivo em código C equivalente aos comandos de entrada 9

6 Testes realizados **11**
6.1 Descrição dos testes realizados 11
6.2 Exemplos de comandos de entrada e os resultados esperados 11

7 Conclusão **13**
7.1 Resumo dos objetivos alcançados no trabalho prático 13
7.2 Reflexão sobre os principais conhecimentos adquiridos durante o desenvolvimento 13

Chapter 1

Introdução

1.1 Problema que foi resolvido no trabalho prático

Neste trabalho prático, o problema a ser resolvido consiste na implementação de uma ferramenta em Python, utilizando a biblioteca PLY, para interpretar uma linguagem capaz de especificar instruções encontradas em qualquer linguagem de programação. A ferramenta desenvolvida deve ser capaz de ler um arquivo de texto (.ea) contendo uma sequência de comandos de especificação de expressões aritméticas, aplicar esses comandos para calcular o resultado desejado e apresentá-lo ao utilizador no terminal. O programa também pode gerar um arquivo com a respetiva implementação em código C. Caso não seja fornecido um arquivo de entrada, os comandos devem ser lidos diretamente do terminal.

O objetivo principal é criar uma linguagem simples e flexível para especificar expressões aritméticas, permitindo a escrita de instruções de escrita, declaração de variáveis, atribuições, leitura de valores do utilizador, geração de valores aleatórios, ciclos, definição e invocação de funções. A implementação deve ser capaz de reconhecer a sintaxe correta da linguagem, avaliar as expressões aritméticas de forma correta e apresentar os resultados de acordo com os comandos fornecidos.

O problema a ser resolvido envolve o desenvolvimento de um reconhecedor léxico e uma gramática adequada para interpretar os comandos, a construção de uma Árvore de Sintaxe Abstrata (AST) para representar a estrutura dos comandos, a avaliação dos comandos de acordo com as regras da linguagem e a apresentação dos resultados de forma adequada. Além disso, o programa deve ser capaz de gerar um código equivalente em C, permitindo a execução das instruções em outra linguagem de programação.

1.2 Importância e contexto do problema

O problema abordado neste trabalho prático é importante no contexto da programação e da linguagem de especificação de expressões aritméticas. A linguagem de programação é uma ferramenta fundamental para a criação de software e a solução de problemas computacionais. Dessa forma, é essencial compreender e implementar uma linguagem que seja capaz de especificar expressões aritméticas de maneira clara e eficiente.

A importância desse problema reside no facto de que a linguagem de especificação de expressões aritméticas é uma parte essencial de muitas linguagens de programação e sistemas computacionais. Ela permite que os programadores expressem cálculos e operações matemáticas de forma concisa e compreensível. Além disso, a capacidade de interpretar e avaliar expressões aritméticas é fundamental para a execução de algoritmos e a resolução de problemas complexos.

No contexto específico deste trabalho prático, a implementação de uma ferramenta em Python que interpreta uma linguagem de especificação de expressões aritméticas

tem como objetivo fornecer aos alunos uma experiência prática na construção de um reconhecedor léxico, gramática e interpretação de comandos. Isso ajuda a desenvolver habilidades de análise sintática e semântica, compreensão de estruturas de dados e algoritmos, e a familiarização com o processo de compilação ou interpretação de linguagens de programação.

Além disso, o contexto do problema também envolve a realização de código em C a partir da linguagem de especificação de expressões aritméticas. Isso possibilita a transferência e execução dos comandos em outra linguagem, ampliando as possibilidades de utilização e integração com outros sistemas ou aplicações.

1.3 Objetivo geral do trabalho

O objetivo geral deste trabalho é implementar uma ferramenta em Python, utilizando a biblioteca PLY, que interprete uma linguagem de especificação de expressões aritméticas. A ferramenta deve ser capaz de ler um arquivo de texto contendo os comandos de especificação, avaliar as expressões aritméticas e apresentar os resultados ao utilizador no terminal. Além disso, opcionalmente, o programa deve gerar um arquivo em código C equivalente aos comandos fornecidos.

Outro objetivo importante é a realização de um código em C a partir da linguagem de especificação de expressões aritméticas. Isso amplia as possibilidades de utilização dos comandos em outras linguagens de programação e promove a compreensão dos processos de compilação ou interpretação de linguagens.

No geral, o objetivo do trabalho é fornecer aos alunos uma oportunidade de desenvolver habilidades práticas na implementação de uma linguagem de programação simples, mas funcional, que seja capaz de especificar expressões aritméticas e executar as operações correspondentes.

Chapter 2

Metodologia

2.1 Abordagem seguida para resolver o problema

Para resolver o problema proposto, foi adotada a seguinte abordagem:

- Análise léxica: A primeira etapa foi a implementação de um analisador léxico utilizando a biblioteca PLY. O analisador léxico é responsável por receber o arquivo de texto de entrada ou os comandos digitados pelo utilizador no terminal e realizar a divisão do código em tokens. Foram definidos os padrões de expressões regulares para identificar os diferentes tipos de tokens, como palavras-chave, identificadores, números inteiros, strings e operadores.

- Análise sintática: Em seguida, foi implementada a análise sintática utilizando a biblioteca PLY. A análise sintática é responsável por verificar a estrutura gramatical correta dos comandos de acordo com as regras da linguagem. Foi definida uma gramática que descreve a sequência correta dos comandos e suas expressões aritméticas associadas. A partir da gramática, foram criadas as regras de produção e associadas a funções Python que constroem a Árvore de Sintaxe Abstrata (AST).

- Construção da Árvore de Sintaxe Abstrata (AST): Durante a análise sintática, a AST é construída para representar a estrutura hierárquica dos comandos. Cada nó da árvore representa um comando ou uma expressão aritmética, e os nós filhos representam os argumentos ou os operadores associados. A AST é utilizada posteriormente para avaliar as expressões aritméticas e executar os comandos.

- Avaliação de expressões aritméticas: Após a construção da AST, a próxima etapa é a avaliação das expressões aritméticas. Foi implementado um mecanismo recursivo que percorre a AST e realiza a avaliação das expressões, levando em consideração os operadores e operandos correspondentes. Durante a avaliação, as variáveis declaradas anteriormente são acessadas e seus valores são utilizados nas operações aritméticas.

- Execução dos comandos e apresentação dos resultados: Após a avaliação das expressões, os comandos são executados e os resultados são apresentados ao utilizador no terminal. Os comandos de escrita (ESCREVER) exibem os valores na saída padrão, realizando a concatenação de strings e os cálculos das expressões aritméticas. Caso seja necessário, o programa também pode gerar um arquivo em código C com a implementação equivalente aos comandos fornecidos.

Essa abordagem seguida para resolver o problema envolve a utilização de conceitos de análise léxica, análise sintática, construção de AST, avaliação de expressões e

execução de comandos. Com essa estrutura, é possível interpretar a linguagem de especificação de expressões aritméticas e realizar as operações correspondentes de forma adequada.

2.2 Descrição de como a leitura do arquivo de texto foi realizada e como os comandos foram interpretados

2.3 Estrutura da Árvore de Sintaxe Abstrata (AST) utilizada na implementação

Chapter 3

Funcionalidades implementadas

3.1 Listagem e explicação das funcionalidades principais da linguagem de expressões aritméticas implementada

3.2 Funcionalidades adicionais implementadas além dos requisitos mínimos do enunciado

3.3 Explicação de como as funcionalidades foram integradas à linguagem

Chapter 4

Desafios encontrados

4.1 Principais desafios enfrentados durante o desenvolvimento

Durante o desenvolvimento da ferramenta em Python, alguns desafios foram enfrentados. Os principais desafios incluem:

- Análise léxica e sintática: A implementação do analisador léxico e sintático utilizando a biblioteca PLY exigiu um entendimento aprofundado de como definir as regras de análise e como lidar com possíveis conflitos e ambiguidades gramaticais. Foi necessário estudar a documentação da PLY e realizar diversos testes e iterações para garantir que as regras estivessem corretas e a análise fosse feita de forma precisa.

- Construção da Árvore de Sintaxe Abstrata (AST): A criação da AST para representar a estrutura dos comandos e expressões aritméticas também foi um desafio. Foi necessário definir uma estrutura de dados adequada para a AST e implementar a lógica para construí-la durante a análise sintática. Garantir a correta representação hierárquica dos comandos e expressões exigiu um cuidadoso planejamento e verificação.

- Avaliação de expressões aritméticas: A correta avaliação das expressões aritméticas, considerando a precedência e associatividade dos operadores, também apresentou desafios. Foi necessário projetar e implementar um mecanismo de avaliação recursiva que tratasse adequadamente as expressões complexas e fornecesse resultados precisos. Lidar com operadores e funções matemáticas foi um aspecto crítico nesse processo.

- Tratamento de erros: Implementar um tratamento adequado de erros foi um desafio importante. Foi necessário identificar os diferentes tipos de erros que poderiam ocorrer durante a análise e execução dos comandos e fornecer mensagens de erro claras e úteis para orientar o utilizador na correção dos problemas. O tratamento de erros ajudou a melhorar a usabilidade da ferramenta e a facilitar a depuração de erros nos programas escritos na linguagem de especificação de expressões aritméticas.

Superar esses desafios exigiu perseverança, dedicação e um processo iterativo de desenvolvimento, com revisões contínuas e melhorias no código. Através desses desafios, foram adquiridos conhecimentos valiosos e habilidades na implementação de uma linguagem de especificação de expressões aritméticas.

Chapter 5

Resultados

5.1 Apresentação dos resultados obtidos com a implementação da ferramenta

5.2 Explicação de como o programa foi capaz de gerar um arquivo em código C equivalente aos comandos de entrada

Chapter 6

Testes realizados

6.1 Descrição dos testes realizados

6.2 Exemplos de comandos de entrada e os resultados esperados

Chapter 7

Conclusão

7.1 Resumo dos objetivos alcançados no trabalho prático

No geral, os objetivos propostos foram alcançados, resultando em uma ferramenta funcional capaz de interpretar a linguagem de especificação de expressões aritméticas, executar os comandos e apresentar os resultados ao utilizador.

7.2 Reflexão sobre os principais conhecimentos adquiridos durante o desenvolvimento

Durante o desenvolvimento deste trabalho prático, algumas conhecimentos importantes foram adquiridos. Estes conhecimentos podem ser resumidos da seguinte forma:

- Conhecimento da biblioteca PLY: A utilização da biblioteca PLY para implementar o analisador léxico e sintático da linguagem de especificação de expressões aritméticas proporcionou um maior entendimento sobre o processo de análise e interpretação de linguagens formais. Aprendemos a criar as regras de tokenização e gramática necessárias para reconhecer a estrutura da linguagem e manipular corretamente os elementos sintáticos.

- Compreensão de linguagens de programação: Ao implementar uma linguagem de programação simplificada, tivemos a oportunidade de compreender melhor como as linguagens de programação funcionam internamente. Aprendemos sobre os diferentes tipos de instruções, declarações, atribuições e expressões aritméticas, e como esses elementos se relacionam para executar um programa.

- Resolução de problemas complexos: Durante o desenvolvimento, nos deparamos com desafios complexos que exigiram a aplicação de raciocínio lógico e habilidades de resolução de problemas. Aprendemos a decompor o problema em partes menores, identificar possíveis soluções e implementar as melhores abordagens para resolver cada aspecto do problema.

- Boas práticas de programação: Durante o desenvolvimento, buscamos seguir boas práticas de programação, como a modularização do código, o uso de nomes significativos para variáveis e funções, e a organização adequada da estrutura do programa. Essas práticas contribuíram para um código mais legível, manutenível e reutilizável.

- Trabalho em equipa: Este trabalho prático foi realizado em equipa, o que nos permitiu desenvolver habilidades de colaboração e comunicação. Aprendemos a compartilhar ideias, dividir tarefas, resolver problemas em conjunto e trabalhar de forma eficiente como um time.

No geral, o desenvolvimento deste trabalho prático proporcionou um ambiente de aprendizagem prática, onde pudemos aplicar conceitos teóricos em um contexto real. As habilidades adquiridas ao longo do desenvolvimento serão valiosas para projetos futuros, tanto na área de linguagens de programação quanto no desenvolvimento de software em geral.